Hierarchical Role-based Viewing
for
Multi-level Information Security in
Collaborative CAD

Christopher D. Cera
Taeseong Kim
Ilya Braude
JungHyun Han
and
William C. Regli

1

Report Documentation Page		*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **JAN 2004**	2. REPORT TYPE	3. DATES COVERED **00-01-2004 to 00-01-2004**

4. TITLE AND SUBTITLE **Hierarchical Role-based Viewing for Multi-level Information Security in Collaborative CAD**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Department of Computer Science,Drexel University,Philadelphia,PA,19104**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**20**	

Hierarchical Role-based Viewing for Multi-level Information Security in Collaborative CAD

Christopher D. Cera [†]   Ilya Braude [†]   Taeseong Kim [‡]
JungHyun Han [‡]   William C. Regli [†] [*]

[†] Geometric and Intelligent Computing Laboratory   [‡] Computer Graphics Laboratory
Department of Computer Science, College of Engineering   School of Information and Communications Engineering
Drexel University   Sung Kyun Kwan University
Philadelphia, PA 19104   Suwon, 440-746, Korea

Abstract

Information security and assurance are new frontiers for collaborative design. In this context, information assurance (IA) refers to methodologies to protect engineering information by ensuring its availability, confidentiality, integrity, non-repudiation, authentication, access control, etc. In collaborative design, IA techniques are needed to protect intellectual property, establish security privileges and create "need to know" protections on critical features.

This paper provides a framework for information assurance within collaborative design based on a technique we call *Role-based Viewing*. We extend upon prior work to present *Hierarchical Role-based Viewing* as a more flexible and practical approach since role hierarchies naturally reflect an organization's line of authority and responsibility. We establish a direct correspondence between multi-level security and multiresolution surfaces where a hierarchy is represented as a weighted directed acyclic graph. The permission discovery process is formalized as a graph reachability problem and the path cost is used as input to a multiresolution function. By incorporating security with collaborative design, the costs and risks incurred by multi-organizational collaboration can be reduced. The authors believes that this work is the first of its kind to unite multi-level security and information clouding with geometric data, including multiresolution surfaces, in the fields of computer-aided design and collaborative engineering.

1   Introduction

Information assurance (IA) refers to methodologies to protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. In collaborative design, IA is mission-critical. Suppose a team of designers is working collaboratively on a 3D assembly model. Each designer has a different set of security privileges and no one on the team has the "need to know" the details of the entire design. In collaboration, designers must interface with others' components/assemblies, but do so in a way that provides each designer with only the level of information he or she is permitted to have about each of the components. For example, one may need to know the

---

[*]Also of the Department of Mechanical Engineering; Email: `regli@drexel.edu`

exact shape of some portion of the part (including mating features) being created by another designer, but not the specifics of any other aspects of the part. Such a need can also be found when manufacturers outsource designing a sub-system: manufacturers may want to hide some critical information of the entire system from suppliers.

The authors believe that a geometric approach to IA represents a new problem that needs to be addressed in the development of collaborative CAD systems. The approach we develop has many uses visible across several significant scenarios we envision for applying this work:

Protection of sensitive design information: As noted above, designers may have "need to know" rights based on legal, intellectual property, or national security requirements.

Collaborative supply chains: Engineering enterprises outsource a considerable amount of design and manufacturing activity. In many situations, the organization needs to provide vital design data to one partner while protecting the intellectual property of another partner.

Multi-disciplinary design: For designers of different disciplines working on common design models, designers suffer from cognitive distraction when they must interact with unnecessary design details that they do not understand and cannot change. For example, an aircraft wheel well [Callahan and Heisserman, 1996] is a complex and confusing place in which electronics, mechanical, and hydraulics engineers all must interact in close quarters with vast amounts of detailed design data. These interactions could be made more efficient if the design space could be simplified to show each engineer only the details they need to see.

This paper develops a new technique for *Role-based Viewing* [Cera et al., 2004] in a collaborative 3D assembly design environment, where multiple users work simultaneously over the network, and presents a combination of *multiresolution geometry* and *multi-level information security models*. Among various issues in IA, *access-control* is critical for the purpose. We demonstrate the specification of access privileges to geometric partitions in 3D assembly models defined based on the Bell-La Padula model. In our method, the partitioning is used to create variable level-of-detail (LOD) meshes, across both individual parts and assemblies, to provide a *Role-based View* suitable for a user with a given level of security clearance. We achieve these functional capabilities within a system designed for secure, real-time collaborative viewing of 3D models by multiple users working synchronously over the internet on standard graphics workstations.

Aside from digital 3D watermarking, research on how to provide IA to distributed engineering teams, working in collaborative graphical environments, remains a novel and relatively unexplored area. The authors believe that this work represents a unique application of multiresolution surfaces to multi-level information security in computer-aided design and collaborative engineering. The specific contributions of this work include:

Provide a geometric approach to Information Assurance: Our work augments currently practiced access-control techniques in collaborative CAD and PDM systems. Although most of these systems offer access-control facilities, they are often limited to prohibiting access to models and documents and not partitions of geometry.

Develop alternatives to the problem of "all-or-nothing" permissions: The standard method for handling a lack of appropriate permissions is suppression of the sensitive features. This work attempts to highlight some alternatives other than the traditional solution.

Outline the relation between Multi-level Security Hierarchies and Multiresolution Surfaces: We revisit the problem of *Role-based Viewing* in an updated context using role hierarchies. A hierarchy is represented as a weighted directed acyclic graph (DAG), where the permission discovery process is formalized as a graph reachability problem and the path cost is used as input to a multiresolution function.

This paper is organized as follows: Section 2 describes related work from information assurance, collaborative design, and computer graphics communities. Section 3 reviews the specification of security features in the fields of solid modeling and engineering as outlined in Reference and presents *Hierarchical Role-based Viewing*. Section 4 explains the details of our multiresolution security model and outlines its relation to the *Role Hierarchy*. Section 5 describes the implementation of our prototype system, and demonstrates a sample scenario using our approach. Lastly, Section 7 summarizes our results, presents our conclusions, and outlines goals for future research.

2   Related Work

The contributions presented in this paper are related to information assurance, collaborative design, and multiresolution surface generation.

2.1   Information Assurance and Security

Current research on information assurance incorporates a broad range of areas focused on protecting information and information systems by ensuring their availability, integrity, confidentiality, non-repudiation, authentication, and controlling modes of access. Information assurance research, in the context of the CAD domain, has been partially addressed by the computer graphics community through the development of 3D digital watermarking [Praun et al., 1999]. Digital Watermarking is used to ensure that the integrity of a model has been maintained, as well as provide a foundation for proof of copyright infringement. Other areas of research have been in authentication and access-control. We will introduce past and present research on access control methodologies and outline the differences between the varying policies.

There is a clear distinction between authentication and access control services. Authentication services are used to correctly determine the identity of a user. Access control is the process of limiting access to resources of a system only to authorized users, programs, processes, or other systems. Authentication is closely coupled with access control, where access control assumes that users of an information system have properly been identified by the system. If the authentication mechanism of a system has been compromised, then the access control mechanism that follows will certainly be compromised. The primary focus of our work is to articulate an access control policy, specifically for the geometry of a solid model, assuming a robust authentication mechanism has already been established. Access-control literature describes high-level policies on how accesses are controlled, as well as low-level mechanisms that implement those policies.

The common access control policies found in literature are Discretionary, Lattice-Based, and Mandatory Access Control (DAC, LBAC, and MAC respectively). DAC was formally introduced by Lampson [Lampson, 1971] , where essentially the owner of an object has discretion over what users were authorized to access that object. Access broadly refers to a particular mode of operation such as read or write. The owner is typically designated as the creator of an object, hence it is an actual user of the system. This is different from LBAC and MAC, which we will refer to collectively as MAC [Lampson, 1971], where individual users have no discretion over object access. MAC [Bell and La-Padula, 1973] is primarily concerned with the flow of information, thereby enforcing restrictions on the direction of communication channels. For further discussion on access control policies, we refer interested readers to a survey by Sandhu [Sandhu and Samarati, 1994].

Role-Based Access Control (RBAC) is an emerging area of study, and is actively pursued as an augmentation of traditional DAC and MAC. RBAC is an instance of a Multi-Level Security (MLS) framework, which is still an actively pursued area in the database community [Jajodia and Sandhu, 1991, Sandhu and Chen, 1998]. In RBAC, individual users are assigned roles, and the access permissions of an object are also assigned to roles. Therefore the permissions assigned to a role are acquired by the

members associated with it. This additional layer reduces the management of permissions and supports the concepts of least privilege, separation of duties, and data abstraction. RBAC, and its associated components, are an instrument for expressing a policy, and not a policy by itself. For *role-based viewing*, we use a MAC policy embodied within an RBAC framework.

2.2 Collaborative Design

There is a vast body of past work on concurrent engineering and collaborative design. In our view, this research can be [loosely] grouped into two categories which we will call "data centric" and "interaction centric."

Data centric research focuses on collaborative data sharing or knowledge sharing. Historically, research of this kind emerged simultaneously from the engineering, the artificial intelligence, and database communities. Interaction centric approaches deal with the real-time or asynchronous collaboration among people in the design process. This most frequently means real-time, collaborative, multi-user environments. Often these environments would be graphical, or 3D; in other cases, the environment consist of computer-supported cooperative work (CSCW) tools coupled with design systems. Much of the recent work in Collaborative Graphics falls into the latter category.

The subset of existing work most relevant to our efforts is interaction centric, dealing with real-time 3D collaboration and communication. Distributed Virtual Environments (DVEs) [Jayaram et al., 1999, Eriksson, 1994, Macedonia et al., 1994] have been developed for real-time interactions between distributed collaborators in a number of different domains. Immersive environments such as CAVEs [Cruz-Neira et al., 1993] have been developed which also support real-time interaction, but they do not necessarily support collaborative CAD. [Conner et al., 1997] directly addressed the use of distributed VR for collaborative design, but in this work the design data was largely static and not worked on synchronously by multiple users. In each of these cases, the work employed large-scale virtual reality systems.

On the scale team design, where individual users collaborate using more typical computing hardware, empirical study is recently beginning to emerge. SHASTRA is an environment for collaborative visualization and shared multimedia, demonstrated mostly for scientific and medical applications [Anupam and Bajaj, 1993]. The DOME [Pahng et al., 1998, Abrahamson et al., 2000] and FIPER [Rohl et al., 2000, Kao et al., 2003] systems target the integration of software products, and coordination between them over the network, for collaboration among individuals assigned disjoint duties in the product development cycle or across institutional boundaries. These systems support an access-control framework, but do not offer alternatives to the problem of "all-or-nothing" feature suppression when a lack of full permissions exists.

Research efforts on level of detail (LOD) rendering [Hoppe, 1998], view-dependent rendering [De Floriani et al., 2000] and 3D compression [Deering, 1995, Taubin and Rossignac, 1998, Gueziec et al., 1999] often mention the applicability of these techniques to collaborative design. To date, however, the main use of these efforts has been limited to areas such as streaming or transmitting 3D data over the Internet.

2.3 Multiresolution Techniques

Polygon meshes lend themselves to fast rendering algorithms, which are hardware-accelerated in most platforms. Many applications, including CAD, require highly detailed models to maintain a convincing level of realism. It is often necessary to provide LOD techniques in order to deliver real-time computer graphics and animations. Therefore, *mesh simplification* is adopted for efficient rendering, transmission, and various computations. The most common use of mesh simplification is to generate *multiresolution models* or various *levels of detail* (LOD). For example, closer objects are rendered with a higher LOD,

and distant objects with a lower LOD. Thanks to LOD management, many applications such as CAD visualization can accelerate rendering and increase interactivity. A recent survey on mesh simplification can be found in Reference [Luebke, 2001].

The most popular polygon-reduction technique is an *edge collapse* or simply *ecol* (more generally, vertex merging or vertex pair contraction) where two vertices are collapsed into a single one. The issues in *ecol* include which vertices to merge in what order, where to place the resulting vertex, etc. *Vertex split* or simply *vsplit* is the inverse operation of *ecol*. These operations are illustrated in Figure 1(a) and a sequence of operations is illustrated on a sample model given in Figure 1(b).

(a) Edge collapse (*ecol*) and vertex split (*vsplit*) operations. v_t (top) and v_b (bottom) collapse into v_m (middle). The inverse operation involves splitting v_m back into v_t and v_b.

(b) Sequence of operations on the "socket" model [National Design Repository, 2003].
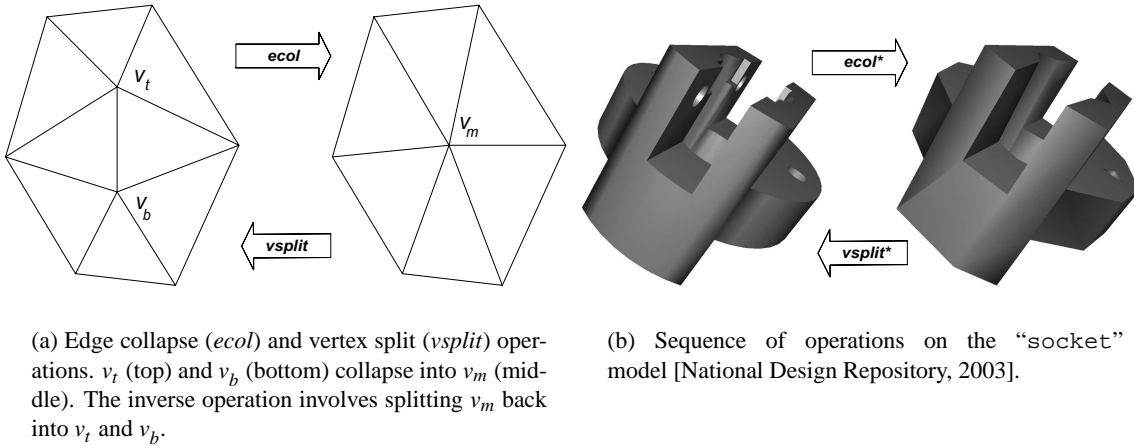
Figure 1: Illustration of Multiresolution techniques.

Hoppe proposed *progressive mesh* (PM) [Hoppe, 1996], which consists of a coarse base mesh (created by a sequence of *ecol* operations) and a sequence of *vsplit* operations. Applying a subset of *vsplit* operations to the base mesh creates an intermediate simplification. The *vsplit* and *ecol* operations are known to be fast enough to apply at runtime, therefore supporting dynamic simplification.

3   Role-based Viewing

In the context of 3D design, a *model M* is a description of an artifact, usually an individual part or assembly, in the form of a solid model. A true collaborative engineering environment enables multiple engineers to simultaneously work with *M*. The engineers (designers, process engineers, etc) correspond to a set of *actors* $A = \{a_0, a_1, \ldots, a_n\}$, each of which has associated with it a set of *roles*. Roles, $R = \{r_0, r_1, \ldots, r_m\}$, define access and interaction rights for the actors. For example, actor a_3 might have associated with it roles r_{20}, r_{23}, and r_{75}—this entitles them to view (and perhaps change) portions of *M* associated with these roles. Portions of *M* not associated with these roles, however, might be "off limits" to actor a_3. This section will build on the results of Reference [Cera et al., 2004], where *Role-based Viewing* was developed in the context of distributed collaborative CAD, by introducing role hierarchies and their relation to multiresolution surfaces.

We formulate the problem of *role-based viewing* in the following subsections by developing:

- **Actor-Role Framework:** a general RBAC framework for describing actors and roles within a collaborative-distributed design environment. This framework uses a hierarchical graph to capture role-role relationships and create a relation between actors and roles.

- **Model-Role Framework:** an associative mapping from roles to topological regions on models. These regions capture the *security features*, F, of a 3D model—relating how a point, patch, part, or assembly can be viewed by actors with given roles.

- **Hierarchical Role-Based Viewing:** an algorithm to generate a *role-based view* given an actor a, his/her set of roles, the role hierarchy (RH), a model M, and its set of security features. A *role-based view* is a tailored 3D model which is customized for actor a based on the roles defining a's access permissions on the model. In this way, the role-based view model does not compromise sensitive model information which a is not allowed to see (or see in detail). This is accomplished using a mesh simplification technique to generate the *role-based view*.

3.1   Actor-Role Security Framework

Our security framework is based on an adaptation of role-based access control, as developed in the information assurance and security literature [Sandhu et al., 1996], to the collaborative design problem. We focus on the relation between actors, their roles and the solid model geometry. This is in contrast to other work on access control in collaborative CAD which has focused mainly on database synchronization/transaction issues [Bancilhon et al., 1985].

Representing Actors and Roles   We define a hierarchical RBAC framework where:

1. **Entities** include a set of actors, $A = \{a_0, a_1, \ldots, a_n\}$ and a set of roles $R = \{r_0, r_1, \ldots, r_m\}$;

2. **Actor-Role Assignment**, AR, is a relation (possibly many-to-many) of actors to roles: $AR \subseteq A \times R$;

3. **Role Hierarchy**, RH, captures the relationships among the roles. For example, the permissions entailed by role r_{75} might be a superset of those entailed by r_{23}. Hence, the role hierarchy is a *weighted*, directed acyclic graph (DAG), $RH = (R, H)$, where $H \subset R \times R$ is the hierarchical set of relationships (edges) among the roles in R. This creates a partial order on R, hence (in the example above) if $< r_{23}, r_{75} > \in H$ then $r_{23} \prec r_{75}$. The weight of each edge in H is given by the real-valued function $w : H \to [0, 1]$.

An example of this RBAC framework is given in Figure 2. For the remainder of this paper, we focus on *read permission* granted by a given set of roles. Rather than "all or nothing" read permissions, our objective is to assign a "degree of visibility" to features of a model based on an actor's roles. Using this formulation, we show how one can implement a Bell-La Padula-based [Bell and La-Padula, 1973] security model for collaborative viewing of CAD data.

Visibility	a_0	a_1	a_2	a_3
r_0	●			
r_1		●		
r_2			●	
r_3				●

(b) An example weighted Role Hierarchy.
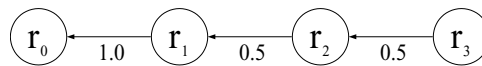
(a) An example Actor-Role assignment matrix.

Figure 2: Example Actor-Role (AR) and Role Hierarchy (RH) assignments.

Example. Using the simple actor-role assignment matrix and role hierarchy from Figure 2, we can compute the degree of visibility to each actor for a model assigned to a specific role. To implement the Bell-La Padula [Bell and La-Padula, 1973] model, we need to compute visibility in such a way as to guarantee that the role (e.g., security clearance) of someone receiving a piece of information must be at least as high as the role assigned to the information itself. In this way, a CAD model classified as "Secret" can only be viewed by those with a "Top-Secret" or "Secret" classification, but not viewed by someone with only a "Confidential" level of access. Figure 3 illustrates this example.
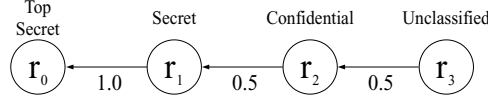

Figure 3: An example weighted Role Hierarchy with associated labels.

3.2 Model-Role Security Framework

Let M be a solid model of an artifact (part, assembly, etc.) and let $b(M)$ represent the boundary of M. In this context, the **Model-Role Assignment**, MR, is a relation (possibly many-to-many) assigning roles to points on the surface of the model: $MR \subseteq b(M) \times R$, where each point on $b(M)$ has at least one role (i.e., $\forall p \in b(M), \exists r \in R$ such that $< p, r > \in MR$). In this way, each point on the surface of the solid model M has associated with it some set of access rights dependent on the roles associated with it.

In practice, it is impractical to assign roles point-by-point to the $b(M)$. Hence, we define a set of *security features*, $F = \{f_0, f_1, \ldots, f_k\}$, where each f_i is a topologically connected patch on $b(M)$ and $\bigcup F = b(M)$; and each f_i has a common set of role assignments. Therefore, the Model-Role assignment can be simplified to be the relation associating security features with access roles: $MR \subseteq F \times R$.

Example. Let M be a solid model; let $F = \{f_1\}$, where $f_1 = b(M)$ (i.e., the entire boundary is one security feature). If $MR = \{< f_1, r_0 >\}$ (where r_0 is from the previous example in Figure 3), then we can see the resultant model for r_0 depicted in Figure 4.
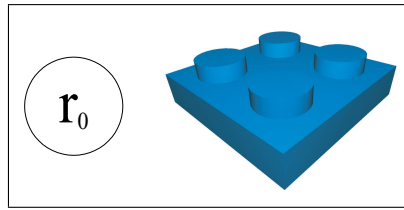
Figure 4: An example part with one security feature where $b(M)$ is assigned to r_0.

3.3 Hierarchical Role-Based Viewing

The issue now is that, for a given actor a, what portions of the model M that he/she can see will depend on their associated roles and the security features of the model. Depending on their permissions, a new model, M', must be generated from M such that the security features are not shown or obfuscated based on the actor's roles. If their roles give them permission to see certain features (i.e., mating features),

then the resulting model includes the features with the same fidelity as in M; if not, the features must be obfuscated in such a way as to hide from a what a does not have the right to see. Hence, the *role-based view* generation problem can be stated as follows:

Problem   Given a set of roles and their relationships (R and RH); a solid model and its security features (M, F, and MR); and an actor (a and AR), determine the appropriate view M' of model M for actor a.

   We propose a solution based on the use of multiresolution meshes, as follows:

1. Convert solid model M to a high-fidelity mesh representation;

2. Based on F, determine which facets belong to each security feature, f;

3. For each security feature f, do:

    (a) If the intersection of actor a's roles and f's roles is non-empty, then add the facets associated with f to M';

    (b) If actor a's roles **do not** intersect the roles of f, determine (using RH) how much of f they are allowed to see and create a set of modified facets to represent f for inclusion in M'.

4. Clean up the resulting M' so that boundaries of the f_i's are topologically valid.

5. Return M'.

 There are three research problems we address:

1. How does the role-hierarchy RH relate to the degree of visibility?

    We show how the weighted DAG that comprises RH can be used to implement a number of useful security policies by making the model quality a function of the "path cost" among roles in RH.

2. How to modify the facets for each f_i based on RH?

    Our approach is to use a security policy (based on Bell-La Padula) associated with the role hierarchy RH to determine how to modify the model. In some cases, policy will dictate degradation of the model fidelity; in other cases, the security features may be completely deleted or replaced with a simple convex hull or bounding box as in Reference [Cera et al., 2004]. To accomplish this, we employ multiresolution meshes: model fidelity will be preserved to the degree the actor's rights allow it. The result is a mesh appropriate for viewing by the actor a.

3. How to ensure that the resulting regions form a topologically valid model?

    Deforming the model feature by feature may result in topological regions of facets in M' that are mis-aligned or aesthetically unpleasing. Cracks and occlusion can be avoided by preserving the boundary edges during simplification.

Example.   This example shows a model M whose surface is described by one security feature f_0. Given the role-hierarchy from Figure 3, and four actors, a_0, a_1, a_2, and a_3 with their AR shown in Figure 2. Figure 5 shows the four different views of model M they each see. Given the AR, RH, and MR assignments, we can derive the direct *actor $\times$ feature* mappings. Figure 6 gives the direct mappings specified implicitly by the AR, RH, and MR given in Figures 2(a), 2(b), and 5 respectively. The two MR assignments that are not shown are $f_1 \in r_1$ and $f_2 \in r_2$. It is important to note that, similar to inheritance found in most object-oriented programming languages, a_0 cannot see f_1 or f_2 even though it is the base role for sub-roles r_1, r_2, and r_3. Hence an inheritance relation allows a child to inherit the permissions of the parent, but nothing is implied in the other direction.
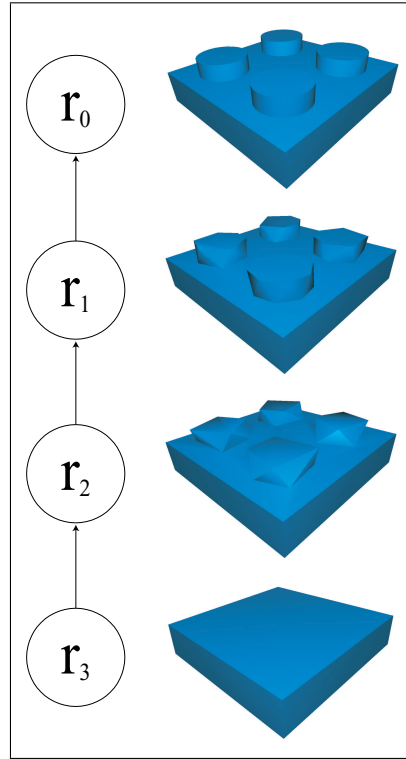
Figure 5: An example part with one security feature (f_0) consisting of $b(M)$ assigned to r_0, a set of actors assigned to roles, and their corresponding set of secure models.

	f_0	f_1	f_2
a_0	1.0	n/a	n/a
a_1	1.0	n/a	n/a
a_2	0.5	1.0	n/a
a_3	0.25	0.5	1.0

Figure 6: The direct permission mappings derived from the *AR*, *RH*, and *MR* relations given in Figures 2(a), 2(b), AND 5 respectively.

4   Technical Approach

We combine techniques from solid modeling and computer graphics to provide a secure collaborative environment which supports real-time design. In this section we describe how to modify and configure Hierarchical RBAC to support our multiresolution security model. We describe the problems, algorithms employed, and final considerations.

4.1 Hierarchical RBAC Policy

Since RBAC is a means of articulating policy rather than a policy by itself, an actual policy is necessary. We wish to adopt a policy similar to the classical MAC model [Bell and La-Padula, 1973]. This is defined in terms of the following axioms using λ to return the security level of either an actor or a feature:

1. **Simple Security Property** - Actor a can read feature f iff $\lambda(a) \geq \lambda(f)$. This is also known as the **read-down** property.

2. **Liberal $*$-Property** Actor a can write feature f iff $\lambda(a) \leq \lambda(f)$. This is also known as the **write-up** property.

There are many variations of the $*$-property, but we will focus on the **simple security property** which essentially states that the clearance of a person receiving a piece of information must be at least as high as the classification of the object. Details on a formal construction of MAC in RBAC have been presented by Osborn [Osborn et al., 2000].

Hierarchical RBAC is a natural means for structuring roles that reflect an organization's lines of authority and responsibility [Sandhu et al., 1996]. The main distinction between our approach and the generic RBAC frameworks found in literature, is that *we also allow permissions to be modified through the role hierarchy*. Typically permissions (i.e., an object and a permissible operation) are associated with every combination of *object* $\times$ *role*. Since our read permissions are specified by a degree of visibility value, an inheritance relation can further refine this value. An inheritance relation is a binary relation ($parent, child$), where the child inherits permissions from the parent based upon a multiplicative weight w. For instance: $w = 1.0$ preserves the parents permissions exactly, while $w = 0.5$ will reduce the degree of visibility by half for all inherited objects. By transitivity, this weighted factor applies to all inherited objects specified in the role hierarchy.

Intuitively, it might appear that we're breaking the **simple security property** by allowing some actors to view objects that they normally would not be able to see. This is not the case, and instead should be viewed as transforming one object into a new object that is permissible. Hence, our model still adheres to the **simple security property**.

Given an actor (a) and a feature (f), the test to determine if a has permissions on f is equivalent to computing graph reachability among all possible pairs of roles assigned to both a and f. We will use R_a to denote the set of roles assigned to a, and R_f for the set of roles assigned to f. If any role in R_a is reachable from any other role in R_f (i.e., there exists a path), then the sum of all weights along the path yields the degree of visibility for that path. We will use a reachability function to return the set of all roles reachable from a given role. This may reveal several paths, hence the resultant degree of visibility for a will be chosen as the maximum. We denote the function that returns the maximum degree of visibility for a on f as $\alpha(a, f)$. The result of this function can be computed once, stored, and re-used until an existing role assignment (*AR* or *MR*) is modified. The degree of visibility is then used as a parameter to another function, $degradeResolution(a, f)$, which degrades the fidelity of a feature depending on an actors permissions.

$\alpha(Actor\ a, Feature\ f)$
1: $R_a = \{roles(a)\};$
2: $R_f = \{roles(f)\};$
3: **if** $R_a \cap R_f\ ! = \emptyset$ **then**
4:    return 1;
5: **else**
6:    // Check *RH* for reachability from r_a to r_f
7:    $DoV = 0.0$
8:    **for all** $r_a \in R_a$ **do**
9:      **for all** $r_f \in R_f$ **do**

10:          **for all** $p \in \{paths(r_a, r_f)\}$ **do**
11:              $DoV = MAX(DoV, path\_cost(p));$
12:          **end for**
13:        **end for**
14:     **end for**
15:     return DoV;
16: **end if**

   $display\_feature(Actor\ a, Feature\ f)$
1: **if** $\alpha(a, f) == 1.0$ **then**
2:     return f;
3: **else**
4:     $f' = degradeResolution(f, \alpha(a, f));$
5:     return f';
6: **end if**

4.2   Generation of Multi-level Security Models

For part/component/assemblies with regions that need to be secured, multiresolution techniques are employed to provide various levels of detail. Although the original (highest) resolution version of a model might be a breach for some actors, lower resolution LODs will be sufficiently secure to transmit to those actors. In addition to purely geometric multiresolution techniques, Shyamsundar and Gadh have developed a framework for representing different levels of detail for geometric feature
data [Shyamsundar and Gadh, 2001, Shyamsundar and Gadh, 2002]. Our security model could be used in conjunction with this feature LOD representation, but an automatic simplification algorithm needs to be developed.

   Mesh simplification techniques include either vertex decimation, vertex clustering, or edge contraction. Choosing a specific simplification technique among the breadth of candidates is application dependent. To address the demands of an interactive collaborative design environment, we outline several issues which are critical for simplification:

   1. *speed*: As the number of component/assemblies in a session increases, the simplification becomes the bottleneck. We need an algorithm capable of drastic simplification in the least amount of time.

   2. *dynamic*: Dynamic simplification provides a continuous spectrum of detail so an appropriate model can be selected at runtime. We do not wish to store all possible LODs within the model repository. Therefore a dynamic simplification will be ideal.

   3. *topology preserving*: To produce the most realistic simplification the original model's topology should be preserved. In addition, progressive meshes [Hoppe, 1996] are incompatible with topology modifying simplification and this technique will be useful for network transmission in a multiuser environment.

   4. *boundary preserving*: The boundary of objects should be preserved in order to distinguish objects from one another. Inadvertent occlusion and cracks may result if we relieve this constraint.

   5. *view-independence*: The viewer receives 3D model information therefore the simplification should also support this.

   Given our requirements, Quadric Error Metrics [Garland and Heckbert, 1997] (QEM) is an obvious candidate. QEM provides drastic simplification, capable of progressivity, in a reasonably small amount of time. This algorithm also produces a result that is realistic and recognizable as a simplified variant of the original model. One issue is the algorithms dependence upon a threshold value. In the rare case

that the threshold value is as large as the model itself, then the algorithm runs in $O(n^2)$. An alternative approach is to compute an optimal threshold adaptively [Erikson and Manocha, 1999].

We have proposed using an automatic simplification technique to degrade the fidelity of a model enough to satisfy the access-control requirements of a collaborative design session. An automatic technique cannot be proven to sufficiently degrade the model enough to be secure in all environments. The process can be supplemented by adopting a form of user-guided simplification [Li and Watson, 2001, Kho and Garland, 2003]. User-guided simplification is a means of supervising the simplification by editing the order of *ecol* performed during simplification, selecting regions where more or less simplification is necessary, or directly manipulating the vertex hierarchy. A side effect is that these simplification parameters need to be stored with the model, since these cannot be automatically derived.

QEM simplification can be configured to either maintain or modify the topological genus of a model. In a multi-user CAD server, progressive meshes (PM) [Hoppe, 1996] can be useful for the transmission of CAD models. If PM is used, and if the removal of holes yields a more secure version of a particular model, then genus-reduction techniques must be employed since standard PM is not compatible with topology-modifying simplification.

Cracks and occlusion must be avoided for continuous and adjacent regions of a part that are simplified independently. If a single part is partitioned into two or more regions, and each region has a different model-role assignment, then the regions will be simplified at different levels of detail. If boundary edges of the mesh are not preserved, then possibly cracks and self-occlusion will result. As a simple example, Figure 7(a) gives an example of a gear tooth that has a different model-role assignment than the rest of the gear. If this tooth is simplified without preserving boundary edges, then, as in Figure 7(b), cracks occur between the regions and self-occlusion results.
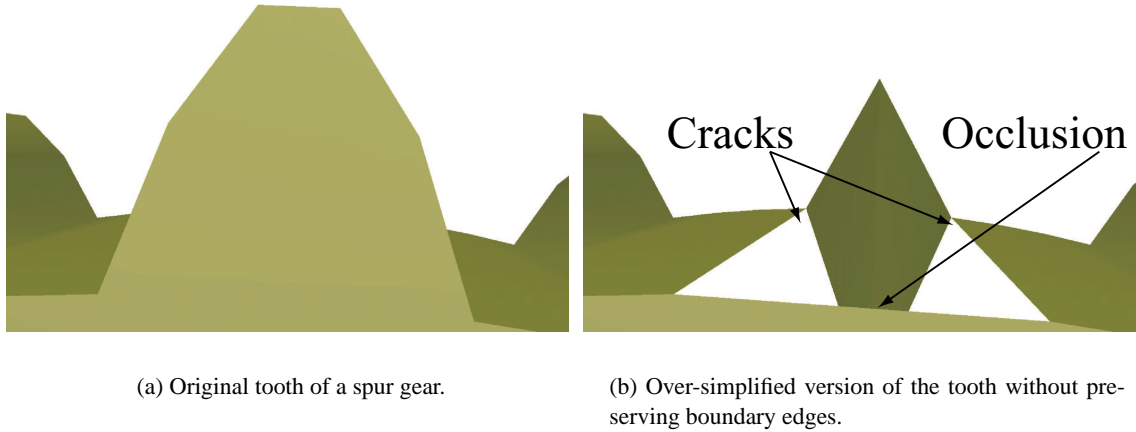

(a) Original tooth of a spur gear.   (b) Over-simplified version of the tooth without preserving boundary edges.

Figure 7: Illustration of cracks and self-occlusion that can occur as a result of simplification.

5   Realization of Approach

The FACADE system is a multi-purpose CAD framework that supports numerous modes of functionality implemented as modules. Its most basic component is a 3D model viewer that supports standard camera navigation operations and the ability to view models using different shading algorithms or as a wireframe. FACADE's design allows instances of the system to be compiled with or without a particular module.

The first module is a light design module which enables several basic tasks such as: selection of a part, component, assembly, or other selectable entity; applying affine transformations on a part; adding an alpha channel to a part for transparency; decomposing a part into multiple parts; specifying a set of parts as an assembly; manipulating control points on parametric surfaces (eg. bezier patches, splines, and NURBS). This subsystem is a prerequisite for most of the remaining modules described below.

The next module provides a semantic authoring interface as developed in Reference [Kopena et al., 2004]. This interface allows a designer to specify the *function* of parts, components, and assemblies, as well as the *flow* of inputs/outputs from one function to another. The module first makes a query, over the network, to the OwlJessKB reasoner asking for the ontological elements that it can use for the authoring of the *function* and *flow* semantics. After annotation is complete, the module provides the ability to save the semantic feature description to an OWL file which will be used during subsequent steps in the conceptual design phase.

The next module, which is at the focus of this paper, is the security authoring interface. This module provides an interface which allows a designer to assign *role-based viewing* parameters to a part, component, assembly, or semantic features that can be saved and later re-loaded. The designing stage allows a designer to assign a {label, permission}-tuple to parts, assemblies, or individual facets. The normalized permissions $[0.0-1.0]$ were used to indicate a percentage of the features to be suppressed from the original model. In situations where the result is not sufficiently secure, a supervised technique, such as user-guided simplification can be used.

When a designer requests a model, they must first declare their identity so all direct role associations can be retrieved and implied associations, from *RH*, can be derived. Based upon the roles associated with a designer and the model features, a *role-based view* is generated. We used a single administrative account to modify permissions in the model repository. There are numerous administrative configurations which have been presented by Sandhu [Sandhu et al., 1999]. The goals and constraints of the collaboration will dictate how comprehensive the role administration requirements should be.

We have implemented our own topology-preserving QEM-based simplification algorithm. For the experiments in this paper, we chose to collapse only vertex pairs which are connected by an edge. The simplification algorithm is passed each tessellated and triangulated part, or connected region of a part with an equivalent {label, permission} set of tuples. Since these regions are disjoint, they can be simplified and transmitted in parallel.

The last module in FACADE enables the network client interface that can talk to a FACADE server. The server works in conjunction with the security module to provide *role-based views* to clients which do not have permissions to manipulate or view a model, or its semantic features, at full resolution. The server maintains consistency throughout all connected clients by sending rejection messages to clients when a design operation they have performed conflicts with the operation of another client. The server supports both "thin" and "fat" FACADE network clients and their corresponding protocols. The thin clients understand the Remote Frame Buffer (RFB) protocol [Richardson and Wood, 1998]. The fat clients understand an unpublished text-based protocol which sends only design transformation information after the initial model is sent.

The FACADE framework has been designed for maximum portability across all platforms. It has been tested and simultaneously developed in Solaris/SunOS (Sun CC/GNU g++), Linux (GNU g++), and Windows 2000 (Microsoft Visual C++) operating systems. It is implemented in C++ using OpenGL as the graphics rendering library. An OpenGL *canvas* can be displayed using either GLUT or Java via

the JNI interface. The network socket libraries use BSD-style sockets under Unix-based derivatives and Winsock2 under Windows. The multi-threaded code uses POSIX threads (pthreads) under Unix-based derivatives and Windows Threads under Windows.

6 Example: Computer Mouse Assembly

We show how role hierarchies can be used to instantiate a multi-level information security model on an electro-mechanical assembly. Unlike previous work in role-based viewing [Cera et al., 2004], this example demonstrates how the weighted role hierarchy affects the collaboration space. We present this as a more flexible and practical approach since role hierarchies naturally reflect an organization's line of authority and responsibility.

In the following example scenario, six actors are granted permission to view and modify a mouse assembly at some level of abstraction. Each actor is assigned a label and assigned a role from the role hierarchy. The set of actors is partitioned into three groups based on the nature of the design work: electrical, mechanical, and ergonomic. Individual parts, regions of parts, or other feature information are assigned labels and grouped into one of the hierarchies. These labels and descriptions are given in Figure 8.

Label	Actor	Hierarchy
a_0	Supervisor	All
a_1	Electrical Engineer	Electrical
a_2	Electrical Observer	Electrical
a_3	Mechanical Engineer	Mechanical
a_4	Mechanical Observer	Mechanical
a_5	Ergonomics Engineer	Ergonomic

(a) Description of actors.

Label	Feature	Hierarchy
f_0	Cable	Electrical
f_1	Circuit Board	Electrical
f_2	Cable/Board Interface	Electrical
f_3	Ball	Mechanical
f_4	Ball Housing	Mechanical
f_5	Lower Casing	Ergonomic
f_6	Buttons	Ergonomic

(b) Description of features.

Figure 8: Actor $\times$ feature labels and descriptions for the mouse assembly example.

Each of the lead engineers will be given full permissions to their respective subsystems. They are each given a subordinate, or observer, who is in training for this design. The sub-roles created for this purpose will be called observer roles. The engineers of one particular hierarchy will also need some level of viewing permissions to the other hierarchies, especially at the interface features. These roles will be called *interface roles*. The roles for the electrical, mechanical, and ergonomic hierarchies are given as r_e, r_m, and r_s respectively. We'll use a second subscript to denote the role's position in the hierarchy where 0 is labeled as the lead roles, 1 is labeled as the observer roles, and 2 is for the interface roles. This actor-role assignment matrix is given in Figure 9(a). The observer and interface roles are given less viewing privileges than the lead roles. The observer roles are given half the degree of visibility as the lead roles, and the interface roles are given half the degree of visibility as the as the observer roles. This weighted hierarchy is depicted in Figure 9(b). Figure 9(c) gives the complete set of model-role assignments for the mouse assembly. In this example, one clear advantage of the role hierarchy is that model-role assignments exist for only lead engineers and the subordinate roles inherit those permissions. Using $\alpha(a, f)$, Figure 9(d) gives the degree of visibility values computed for every *actor* $\times$ *feature*. These values are implicit in the *AR*, *RH*, and *MR* structure.
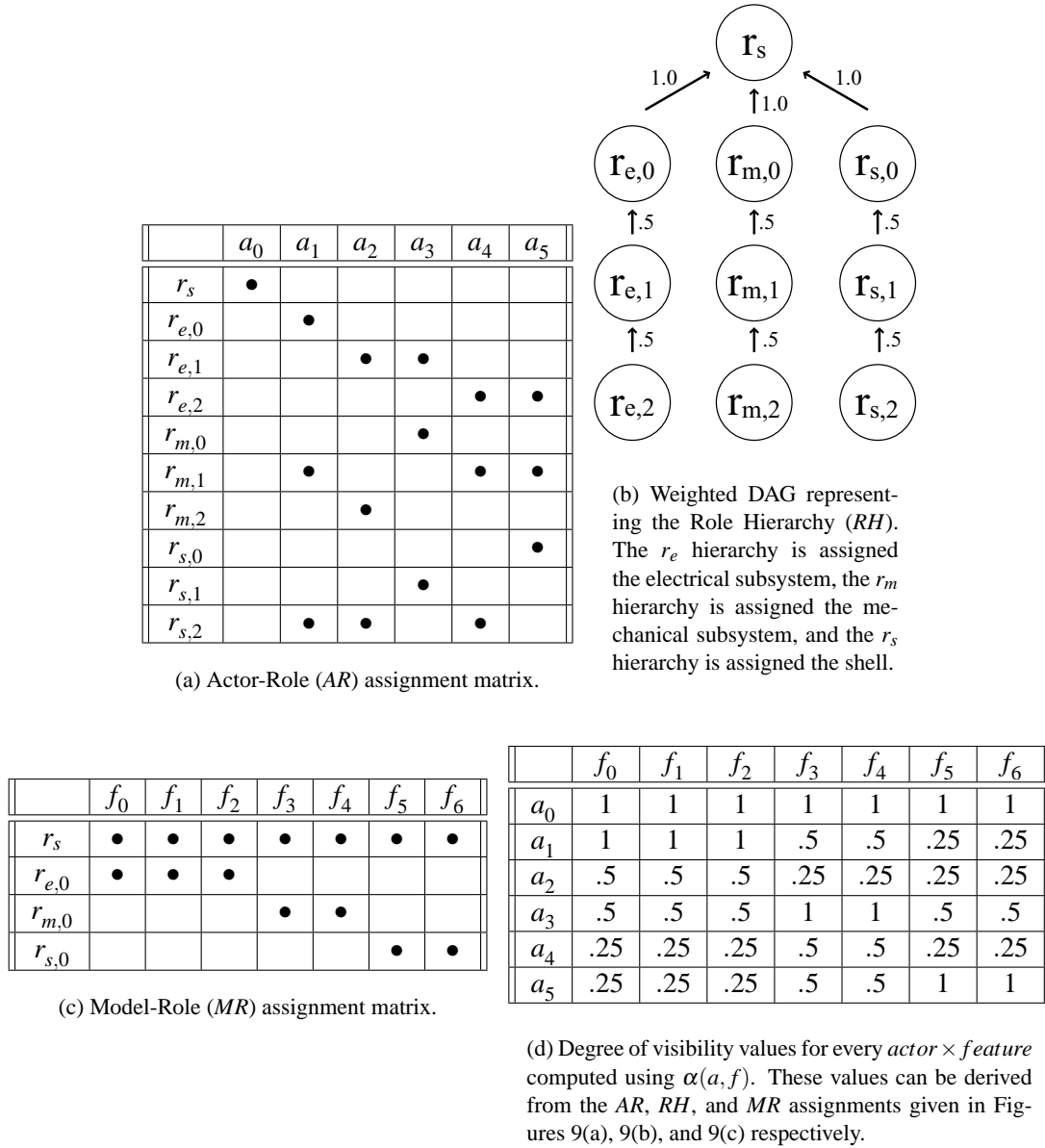
14

	a_0	a_1	a_2	a_3	a_4	a_5
r_s	•					
$r_{e,0}$		•				
$r_{e,1}$			•	•		
$r_{e,2}$					•	•
$r_{m,0}$				•		
$r_{m,1}$		•			•	•
$r_{m,2}$			•			
$r_{s,0}$						•
$r_{s,1}$				•		
$r_{s,2}$		•	•		•	

(a) Actor-Role (*AR*) assignment matrix.

(b) Weighted DAG representing the Role Hierarchy (*RH*). The r_e hierarchy is assigned the electrical subsystem, the r_m hierarchy is assigned the mechanical subsystem, and the r_s hierarchy is assigned the shell.

	f_0	f_1	f_2	f_3	f_4	f_5	f_6
r_s	•	•	•	•	•	•	•
$r_{e,0}$	•	•	•				
$r_{m,0}$				•	•		
$r_{s,0}$						•	•

(c) Model-Role (*MR*) assignment matrix.

	f_0	f_1	f_2	f_3	f_4	f_5	f_6
a_0	1	1	1	1	1	1	1
a_1	1	1	1	.5	.5	.25	.25
a_2	.5	.5	.5	.25	.25	.25	.25
a_3	.5	.5	.5	1	1	.5	.5
a_4	.25	.25	.25	.5	.5	.25	.25
a_5	.25	.25	.25	.5	.5	1	1

(d) Degree of visibility values for every *actor* $\times$ *feature* computed using $\alpha(a, f)$. These values can be derived from the *AR*, *RH*, and *MR* assignments given in Figures 9(a), 9(b), and 9(c) respectively.

Figure 9: All security associations and derived mappings for the mouse assembly example.

The supervisor (a_s) has unrestricted access to all features of the assembly. The supervisor's role-based view is given in Figure 10(a). Figure 10(b) shows the role-based view for the electrical engineer ($a_{e,0}$) which shows the electrical features in full resolution, the mechanical features in a lower resolution, and the exterior features in an even lower resolution. Figure 10(c) gives the role-based view for the mechanical engineer where the mechanical features are displayed in full resolution, the electrical features in a lower resolution, and the exterior features in even lower detail. Figure 10(d) depicts the

ergonomics engineer's ($r_{s,0}$) role-based view which depicts the interior and exterior in full resolution, but the remaining features are displayed in a low resolution. By using role-based views, designers need not be concerned with unnecessary design details and the protection of sensitive intellectual property can be maintained.

7   Conclusions and Future Work

This paper developed a new technique, *Hierarchical Role-based Viewing*, for multi-level information security in collaborative 3D assembly design. Role Hierarchies naturally reflect an organization's line of authority and responsibility. By incorporating security with collaborative design, the costs and risks incurred by multi-organizational collaboration can be reduced. Aside from digital 3D watermarking, research on how to provide security issues to distributed design, working in collaborative graphical environments, remains a novel and relatively unexplored area. The authors believe that this work is the first of its kind to bring multi-level security to geometric data in the field of computer-aided design and collaborative engineering.

Immediate future work involves using multiresolution techniques directly on the native surface types and examining network configurations to reduce aggregate bandwidth. We are currently extending these techniques to handle B-spline surfaces directly. The motivation for handling these surfaces is to demonstrate that for certain geometry, multiresolution surface techniques will provide a more intuitive simplification result. Crack prevention, permissions on patch boundaries when adjacent patches have different roles, and other issues will need to be addressed. We would also like to give a demonstration of the model on geometric, as well as semantic, feature data.

Our environment has been extended to support synchronous multi-user collaborative CAD. Optimal network configurations can be constructed and "grouping" of the mesh hierarchy can be performed for actors and assigned similar roles. We can take advantage of continuous LOD over a network using a progressive technique, such as Progressive Meshes [Hoppe, 1996]. This results in computing only one mesh hierarchy for an entire set of actors. For further optimization, multicast networks could be used to properly aggregate bandwidth when actors have similar privileges.

References

[Abrahamson et al., 2000] Abrahamson, S., Wallace, D., Senin, N., and Sferro, P. 2000. Integrated Design in a Service Marketplace. *Computer-Aided Design*, 32(2):97–107.

[Anupam and Bajaj, 1993] Anupam, V. and Bajaj, C. L. 1993. Collaborative Multimedia Scientific Design in SHASTRA. In *Proceedings of the first ACM international conference on Multimedia*, pp. 447–456. ACM Press.

[Bancilhon et al., 1985] Bancilhon, F., Kim, W., and Korth, H. F. 1985. A Model of CAD Transactions. In Pirotte, A. and Vassiliou, Y., editors, *VLDB'85, Proceedings of 11th International Conference on Very Large Data Bases*, pp. 25–33. Morgan Kaufmann.

[Bell and La-Padula, 1973] Bell, D. and La-Padula, L. 1973. Secure Computing Systems: Mathematical Foundation and Model. *MITRE Report (MTR 2547) v2*.

[Callahan and Heisserman, 1996] Callahan, S. and Heisserman, J. 1996. A Product Representation to Support Process Automation. In Pratt, M., Sriram, R., and Wozny, M., editors, *Product Modeling for Computer Integrated Design and Manufacture*, pp. 285–295. Chapman and Hall.

[Cera et al., 2004] Cera, C. D., Kim, T., Han, J., and Regli, W. C. 2004. Role-Based Viewing in Secure Collaborative CAD. *Accepted to the Journal of Computer Aided Design*.

[Conner et al., 1997] Conner, B., Cutts, M., Fish, R., Fuchs, H., Holden, L., Jacobs, M., Loss, B., Markosian, L., Riesenfeld, R., , and Turk, G. 1997. An Immersive Tool for Wide-Area Collaborative Design. In *TeamCAD, the First Graphics Visualization, and Usability (GVU) Workshop on Collaborative Design*, pp. 139–143.

[Cruz-Neira et al., 1993] Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. 1993. Surround-screen Projection-based Virtual Reality: the Design and Implementation of the CAVE. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 135–142. ACM Press.

[De Floriani et al., 2000] De Floriani, L., Magillo, P., Morando, F., and Puppo, E. 2000. Dynamic View-dependent Multiresolution on a Client–Server Architecture. *Computer-Aided Design*, 32(13):805–823.

[Deering, 1995] Deering, M. 1995. Geometry Compression. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 13–20. ACM Press.

[Erikson and Manocha, 1999] Erikson, C. and Manocha, D. 1999. GAPS: General and Automatic Polygonal Simplification. In *Symposium on Interactive 3D Graphics*, pp. 79–88.

[Eriksson, 1994] Eriksson, H. 1994. MBONE: The Multicast Backbone. *Communications of the ACM*, 37(8):54–60.

[Garland and Heckbert, 1997] Garland, M. and Heckbert, P. S. 1997. Surface Simplification Using Quadric Error Metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 209–216. ACM Press/Addison-Wesley Publishing Co.

[Gueziec et al., 1999] Gueziec, A., Taubin, G., Horn, B., and Lazarus, F. 1999. A Framework for Streaming Geometry in VRML. *IEEE Computer Graphics and Applications*, 19(2):68–78.

[Hoppe, 1996] Hoppe, H. 1996. Progressive Meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 99–108. ACM Press.

[Hoppe, 1998] Hoppe, H. 1998. Efficient Implementation of Progressive Meshes. *Computers and Graphics*, 22(1):27–36.

[Jajodia and Sandhu, 1991] Jajodia, S. and Sandhu, R. 1991. Toward a multilevel secure relational data model. In *Proceedings of the 1991 ACM SIGMOD international conference on Management of data*, pp. 50–59. ACM Press.

[Jayaram et al., 1999] Jayaram, S., Jayaram, U., Wang, Y., Tirumali, H., Lyons, K., and Hart, P. 1999. VADE: a Virtual Assembly Design Environment. *IEEE Computer Graphics and Applications*, 19(6):44–50.

[Kao et al., 2003] Kao, K. J., Seeley, C. E., Yin, S., Kolonay, R. M., Rus, T., and Paradis, M. 2003. Business-To-Business Virtual Collaboration of Aircraft Combustor Design. In *ASME Design Engineering Technical Conferences, Computers and Information in Engineering Conference (DETC2003/CIE-48282)*, Chicago, Illinois. ASME Press.

[Kho and Garland, 2003] Kho, Y. and Garland, M. 2003. User-guided Simplification. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pp. 123–126. ACM Press.

[Kopena et al., 2004] Kopena, J. B., Cera, C. D., and Regli, W. C. 2004. Engineering Design Knowledge Management Based on Conceptual Design. Submitted to the First International Conference on Design Computing and Cognition.

[Lampson, 1971] Lampson, B. 1971. Protection. In *Proceedings of the 5th Annual Princeton Conference on Information Sciences and Systems*, pp. 437–443, Princeton University.

[Li and Watson, 2001] Li, G. and Watson, B. 2001. Semiautomatic Simplification. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pp. 43–48. ACM Press.

[Luebke, 2001] Luebke, D. P. 2001. A Developer's Survey of Polygonal Simplification Algorithms. *IEEE Computer Graphics and Applications*, 21(3):24–35.

[Macedonia et al., 1994] Macedonia, M., Zyda, M., Pratt, D., Barham, P., and Zeswitz, S. 1994. NPSNET: A Network Software Architecture for Large-Scale Virtual Environments. *Presence*, 3(4):265–287.

[National Design Repository, 2003] National Design Repository 2003. National Design Repository, `http://www.designrepository.org`.

[Osborn et al., 2000] Osborn, S. L., Sandhu, R. S., and Munawer, Q. 2000. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies. *Information and System Security*, 3(2):85–106.

[Pahng et al., 1998] Pahng, F., Senin, N., and Wallace, D. 1998. Distribution Modeling and Evaluation of Product Design Problems. *Computer-Aided Design*, 30(6):411–423.

[Praun et al., 1999] Praun, E., Hoppe, H., and Finkelstein, A. 1999. Robust Mesh Watermarking. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 49–56. ACM Press/Addison-Wesley Pub.

[Richardson and Wood, 1998] Richardson, T. and Wood, K. R. 1998. The RFB Protocol version 3.3. Protocol Specification.

[Rohl et al., 2000] Rohl, P. J., Kolonay, R. M., Irani, R. K., Sobolewski, M., Kao, K., and Bailey, M. W. 2000. A Federated Intelligent Product Environment. In *Proceedings of the Eighth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*.

[Sandhu et al., 1999] Sandhu, R., Bhamidipati, V., and Munawer, Q. 1999. The ARBAC97 Model for Role-Based Administration of Roles. *ACM Transactions on Information and System Security (TISSEC)*, 2(1):105–135.

[Sandhu and Chen, 1998] Sandhu, R. and Chen, F. 1998. The multilevel relational (MLR) data model. *ACM Trans. Inf. Syst. Secur.*, 1(1):93–132.

[Sandhu et al., 1996] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. 1996. Role-Based Access Control Models. *IEEE Computer*, 29(2):38–47.

[Sandhu and Samarati, 1994] Sandhu, R. S. and Samarati, P. 1994. Access Control: Principles and Practice. *IEEE Communications*, 32(9):40–48.

[Shyamsundar and Gadh, 2001] Shyamsundar, N. and Gadh, R. 2001. Internet-based Collaborative Product Design with Assembly Features and Virtual Design Spaces. *Computer-Aided Design*, 33(9):637–651.

[Shyamsundar and Gadh, 2002] Shyamsundar, N. and Gadh, R. 2002. Collaborative Virtual Prototyping of Product Assemblies over the Internet. *Computer-Aided Design*, 34(10):755–768.

[Taubin and Rossignac, 1998] Taubin, G. and Rossignac, J. 1998. Geometric Compression Through Topological Surgery. *ACM Transactions on Graphics*, 17(2):84–115.
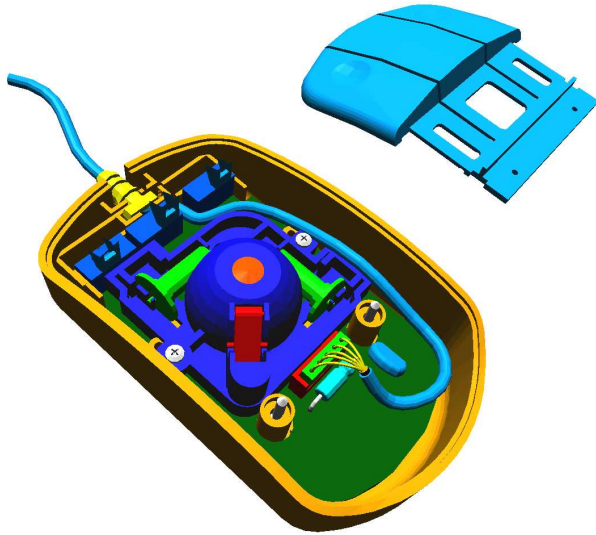
(a) The supervisor's (a_0) full resolution view.
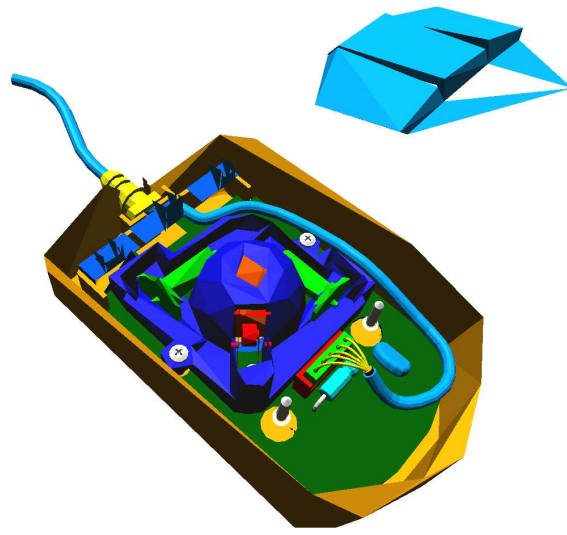
(b) The electrical engineer's (a_1) role-based view. The electrical features are displayed in full resolution, the mechanical features in a lower resolution, and the exterior features in an even lower resolution.
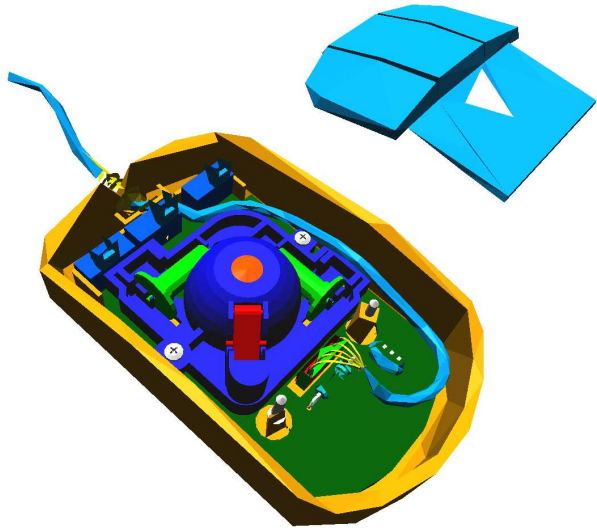
(c) The mechanical engineer's (a_3) role-based view. Mechanical features are displayed in full resolution, the electrical features in a lower resolution, and the exterior features in even lower detail.
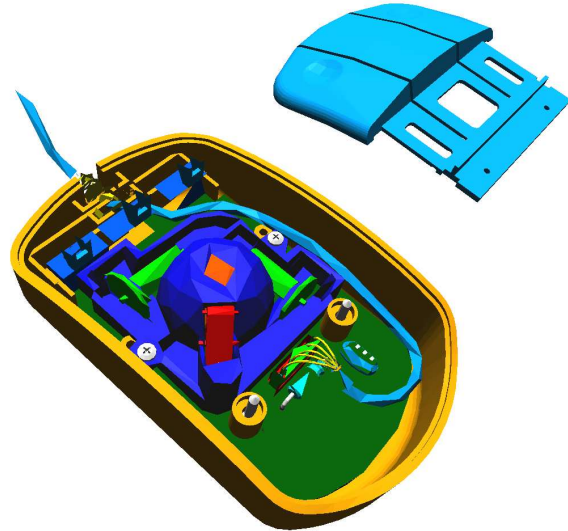
(d) The ergonomic engineer's (a_5) role-based view. This depicts the interior and exterior in full resolution, but the remaining features are displayed in a low resolution.

Figure 10: Role-based views for the mouse assembly example.